

Remarks/Arguments

Applicant has received and carefully reviewed the Office Action mailed October 22, 2003. Claims 1-27 remain pending. Reexamination and reconsideration are respectfully requested.

In paragraph 3 of the Office Action, the Examiner objected to the drawings because the reference character 158 in Figure 7 has been used to designate both the connection between the memory interface and instruction cache tag logic and the connection between I-FLC and the instruction read address control. A proposed drawing correction for Figure 7 and an annotated drawing showing the changes are enclosed herewith for approval by the Examiner. The specification has also been amended to correspond to the proposed reference numeral changes.

In paragraph 4 of the Office Action, the Examiner objected to the drawings because they do not include the following reference signs mentioned in the description: Page 12, lines 14 and 21, element "30" and Page 12, line 20, element "32". The specification has been amended to comply with the corrected drawings.

In paragraph 5 of the Office Action, the Examiner objected to the drawings because they include the following reference signs not mentioned in the description: Figure 7, elements 158, 158b, 192, 175 and 309. As noted above, a proposed drawing correction for Figure 7 and an annotated drawing showing the changes are enclosed herewith for approval by the Examiner. The specification has also been amended to correspond to the proposed reference numeral changes.

Application No. 09/727,744  
Amendment dated January 20, 2004  
Reply to Office Action of October 22, 2003

In paragraph 6 of the Office Action, the Examiner states that the title of the invention is not descriptive. In response, the title has been amended to be more descriptive.

In paragraph 7 of the Office Action, the Examiner objected to claims 11-18 and 20-26. The Examiner noted that there were two claims labeled "2" in the originally filed claims. Since the Office has renumbered the claims, the dependencies of claims 11-18 and 20-26 are now incorrect. Claims 3-27 have been amended to be numbered properly, including the dependencies.

In paragraph 9 of the Office Action, the Examiner rejected claims 1-18 under 35 U.S.C. 102(b) as being taught by Nakayama et al. (U.S. Patent No. 4,788,655). Referring to claim 1, the Examiner states that Nakayama et al. suggest a method for storing a digital value to memory in a pipelined instruction processor, wherein the digital value is read from memory in response to a conditional jump instruction to determine if the condition of the conditional jump instruction is satisfied (citing Nakayama et al.: Abstract; column 1, lines 9-13 and 49-68; column 2, lines 20-50; Figure 1A; Figure 1B; Figure 5A; and Figure 5B). The Examiner also states that Nakayama et al. suggest generating at least one status bit based on the digital value to be stored, the at least one status bit indicating if a predetermined condition of a conditional jump instruction is satisfied (citing Nakayama et al.: Abstract; column 1, lines 9-13 and 49-68; column 2, lines 20-50; Figure 1A; Figure 1B; Figure 5A; and Figure 5B). The Examiner further states that Nakayama et al. suggest storing the digital value and the at least one status bit to memory (citing Nakayama et al.: Abstract; column 1, lines 9-13 and 49-68; column 2, lines 20-50; Figure 1A; Figure 1B; Figure 5A; and Figure 5B).

After carefully reviewing Nakayama et al., Applicants must respectfully disagree. Nakayama et al. do not appear to relate in any way to a method for storing a digital value to memory in a pipelined instruction processor, wherein the digital value is read from memory in response to a conditional jump instruction to determine of the condition of the conditional jump instruction is satisfied, as suggested by the Examiner (Emphasis Added). Furthermore, Nakayama et al. do not appear to generating at least one status bit based on the digital value to be stored, the at least one status bit indicating if a predetermined condition of a conditional jump instruction is satisfied, or storing the digital value and the at least one status bit to memory, as suggested by the Examiner (Emphasis Added).

Applicants note that the terms "pipeline", "branch" and "jump" are never even mentioned in Nakayama et al. The term "condition" does appear in Nakayama et al., but its meaning does not relate in any way to a "condition" of a conditional jump instruction. For example, Nakayama et al. state:

Generally, there are two formats for the binary floating point data, that is, the single-precision format and the double-precision format. In each of the two formats, one binary floating point data is constituted by a sign portion S, an exponent portion EXP and a fraction portion FRAC. With respect to the binary floating point data, the IEEE defines the reserved values as follows.

- 
- |                          |                                     |
|--------------------------|-------------------------------------|
| (1) Not-a-number:        | EXP=All "1" and FRAC.noteq.All "0"; |
| (2) Infinity:            | EXP=All "1" and FRAC=All "0";       |
| (3) Zero:                | EXP=All "0" and FRAC=All "0";       |
| (4) Denormalized number: | EXP=All "0" and FRAC.noteq.All "0". |
- 

When performing the binary floating point arithmetic operation, a condition is set by the sign portion S, a least significant bit (LSB) and the fraction portion FRAC being not equal to "0", and this condition will be referred to as an external condition in the present specification. The word

"external" is used to mean external of an arithmetic logic unit (ALU).

When performing the binary floating point arithmetic operation, the arithmetic processing is performed depending on the external condition of the binary floating point data in a source operand and a destination operand, as will be described later on in the present specification. Accordingly, before performing the binary floating point arithmetic operation, it is necessary to check the external condition of the binary floating point data in the source operand and the destination operand, and furthermore, it is necessary to set the external condition of the binary floating point data, which is obtained as a result of the arithmetic operation, in a status register which can be checked by software.

(Emphasis Added) (Nakayama et al., column 1, lines 35-68). As can readily be seen, Nakayama et al. generate an external condition to identify attributes of the binary floating point data of source and destination operands so that binary floating point arithmetic operations are performed properly. Nowhere do Nakayama et al. disclose or suggest generating at least one status bit that indicates if a predetermined condition of a conditional jump instruction is satisfied, as recited in claim 1. As noted above, the terms "pipeline", "branch" and "jump" are never even mentioned in Nakayama et al. If the Examiner elects to maintain this rejection, Applicants respectfully request that the Examiner specifically point out where in Nakayama et al. each and every element of claim 1 is disclosed.

In view of the foregoing, claim 1 is believed to be clearly patentable over Nakayama et al. For similar and other reasons, dependent claims 2-9 are also believed to be clearly patentable over Nakayama et al. For similar and other reasons, independent claim 10, and dependent claims 11-18 are also believed to be clearly patentable over Nakayama et al.

In paragraph 29 of the Office Action, the Examiner rejected claims 19-22 and 24-27 under 35 U.S.C. § 103(a) as being unpatentable over Watson et al. (U.S. Patent No. 3,573,854) in view of Nakayama et al. The Examiner states that Watson et al. do not teach or suggest: (a) one or more of the conditional jump instructions reading a digital value from memory to determine if the condition of the conditional jump instruction is satisfied; (b) storing a value that includes a digital value and at least one jump status bit; and (c) wherein the current instruction includes an address and a corresponding jump field, the address identifies one of the addressable registers and the corresponding jump field identifies a jump status bit of the at least one jump bits within the identified addressable register.

While Applicants agree with the Examiner that Watson et al. do not teach or suggest these elements, Applicants also believe that Watson et al. do not teach or suggest many other elements of claim 19, including the jump look ahead controller, the tracking logic and the conflict detection logic as recited.

In any event, the Examiner states that Nakayama et al. suggest: (a) one or more of the conditional jump instructions reading a digital value from memory to determine if the condition of the conditional jump instruction is satisfied; (b) storing a value that includes a digital value and at least one jump status bit; and (c) wherein the current instruction includes an address and a corresponding jump field, the address identifies one of the addressable registers and the corresponding jump field identifies a jump status bit of the at least one jump bits within the identified addressable register (citing Nakayama et al.: Abstract; column 1, lines 9-13 and 49-68; column 2, lines 20-50, Figure 1A; Figure 1B; Figure 5A; and Figure 5B).

As noted above, Nakayama et al. do not appear to relate in any way to conditional jump instructions, and in particular, one or more conditional jump instructions that read a digital value from memory to determine if the condition of the conditional jump instruction is satisfied. In addition, Nakayama et al. do not appear to relate in any way to storing a value that includes a digital value and at least one jump status bit (Emphasis Added). Finally, Nakayama et al. do not appear to relate in any way to a system whereby the current instruction includes an address and a corresponding jump field, where the address identifies one of the addressable registers and the corresponding jump field identifies a jump status bit of the at least one jump bits within the identified addressable register (Emphasis Added). As noted above, the terms “branch” and “jump” are never even mentioned in Nakayama et al. As such, Nakayama et al. does not appear to add anything to Watson et al. in this regard.

For these and other reasons, claim 19 is believed to be clearly patentable over Watson et al. in view of Nakayama et al. For similar and other reasons, dependent claims 20-26 are also believed to be clearly patentable over Watson et al. in view of Nakayama et al.

Turning now to claim 27. The Examiner states that Watson et al. suggest generating a jump look-ahead signal that is a function of the selected jump status bit read from the selected address location of the addressable memory, where the identified jump status bit is accessed using the address and the jump field of the current instruction. After careful review, Applicants must respectfully disagree. Nowhere do Watson et al. disclose or suggest a jump status bit, and in particular, a jump status bit that is accessed using the address and the jump field of the current instruction.

Watson et al. state that their invention is primarily useful in the processing of instruction loops. Watson et al. further state that it is well known that the overhead time spend due to an occasional wrong guess at the look-ahead level would be low. However, if this is multiplied by a large number of turns in a program loop, the overhead can be substantial (see Watson et al., column 6, lines 4-8). Watson et al. further state:

The present invention enhances the utility of a look-ahead operation responding to the existence of an instruction which is inserted in the instruction stream immediately preceding the first instruction in the loop. The response to the look-ahead instruction has no effect on the control of the loop. It does, however, require response of the look-ahead system such that the 18th instruction following the look-ahead instruction is a conditional branch for which look-ahead mechanism should provide response to instructions along the branch path rather than continuing further down the instruction list beyond the 18th instruction.

The location of the look-ahead instruction is stored and then used when the look-ahead system has proceeded in its response through the 18 instructions. The response relates only to look-ahead and no to actual control of the program loop.

On the last turn of the program loop, the look-ahead control again returns to the look-ahead instruction. However, when the execution of instruction 115 dictates that the actual program execution should proceed downstream, the condition having been satisfied, means are provided for resetting the look-ahead mechanism, thereby ignoring those instructions fetched under control of the look-ahead mechanism. The look-ahead system is then redirected downstream and responds to downstream instructions thereafter until the next look-ahead instruction is encountered. This response is such that any exit from the loop will cause the look-ahead system to be reset.

(Watson et al., column 6, lines 9-37). As can be seen, Watson et al. suggest inserting a look-ahead instruction into the instruction stream to indicate the beginning of a loop in the instruction stream. The look-ahead instruction then interacts with a look-ahead mechanism, causing the look-ahead mechanism to recognize a loop beginning with the look-ahead instruction and ending with a subsequent conditional branch instruction. The look-ahead mechanism appears to automatically retrieve the instruction block that

includes the look-ahead instruction when approaching the end of the loop (i.e. assumes the loop will continue looping). Note that when the condition of the conditional branch instruction is satisfied, means are provided for resetting the look-ahead mechanism, thereby ignoring those instructions fetched under control of the look-ahead mechanism.

As can be seen, nothing here discloses or suggests a jump status bit, and in particular a jump status bit that is accessed using the address and the jump field of the current instruction, as the Examiner suggests.

The Examiner also states that Watson et al. suggest tracking the addresses of a predetermined number of previous instructions in the pipelined instruction processor and comparing the addresses to the address of the current instruction to generate a series of jump disable signals. Applicants do not see where in Watson et al. “a series of jump disable signals” are suggested.

The Examiner further states that Watson et al. suggest generating a jump early signal using the jump look-ahead signal and the series of jump disable signals, where the jump early signal initiates a conditional jump depending on the value of the jump disable signals. As noted above, Applicants do not see where in Watson et al. “a series of jump disable signals” are suggested. Also, Applicants do not believe Watson et al. disclose or suggest generating a jump early signal using a jump look-ahead signal and a series of jump disable signals.

The Examiner states that Watson et al. do not teach or suggest: (a) storing a digital value and one or more jump status bits that are based on the digital value in each of a plurality of address locations in an addressable memory; and (b) accessing a current instruction, the current instruction having an address and a jump field, the address



Application No. 09/727,744  
Amendment dated January 20, 2004  
Reply to Office Action of October 22, 2003

identifies a selected address location of the addressable memory, and the jump field identifies a selected jump status bit of the selected address location. However, the Examiner states that Nakayama et al. suggest these elements. After careful review, Applicants must respectfully disagree.

As noted above, Nakayama et al. do not appear to relate in any way to conditional jump instructions. In addition, Nakayama et al. do not appear to disclose or suggest the step of storing a value that includes a digital value and at least one jump status bit (Emphasis Added). Furthermore, Nakayama et al. do not appear to disclose or suggest a current instruction that has an address and a jump field, wherein the address identifies a selected address location of the addressable memory, and the jump field identifies a selected jump status bit of the selected address location (Emphasis Added). As noted above, the terms “branch” and “jump” are never even mentioned in Nakayama et al. As such, Nakayama et al. appears to add little to Watson et al. in this regard.

If the Examiner elects to maintain the rejection of claim 27, Applicants respectfully request that the Examiner specifically point out where in Watson et al. and/or Nakayama et al. each and every of these elements can be found. In view of the foregoing, claim 27 is believed to be clearly patentable over Watson et al. in view of Nakayama et al.

In view of the foregoing, it is believed that all pending claims 1-27 are now in condition for allowance. Issuance of a notice of allowance in due course is respectfully

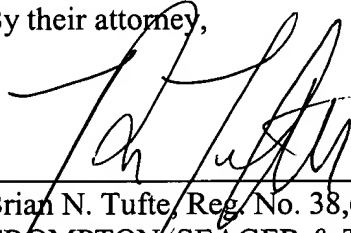
Application No. 09/727,744  
Amendment dated January 20, 2004  
Reply to Office Action of October 22, 2003

requested. If a telephone conference would be of assistance, please contact the undersigned attorney at 612-677-9050.

Respectfully submitted,

Lawrence R. Fontaine et al

By their attorney,

A handwritten signature in black ink, appearing to read "Brian N. Tufte", is written over a horizontal line.

Dated: January 20, 2004

Brian N. Tufte, Reg. No. 38,638  
CROMPTON, SEAGER & TUFTE, LLC  
1221 Nicollet Avenue, Suite 800  
Minneapolis, MN 55403-2402  
Telephone: (612) 677-9050  
Facsimile: (612) 359-9349

**FIG. 7**

